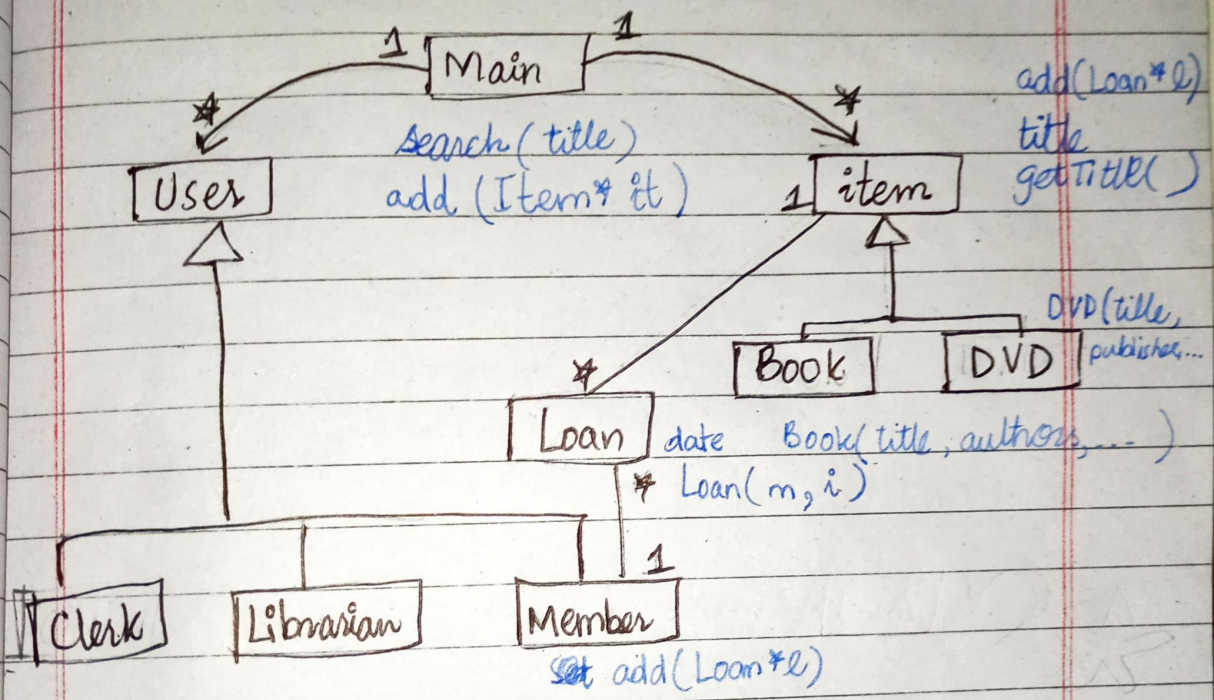
Lecture # 09

Date: Sept 15,25 Day: Monday

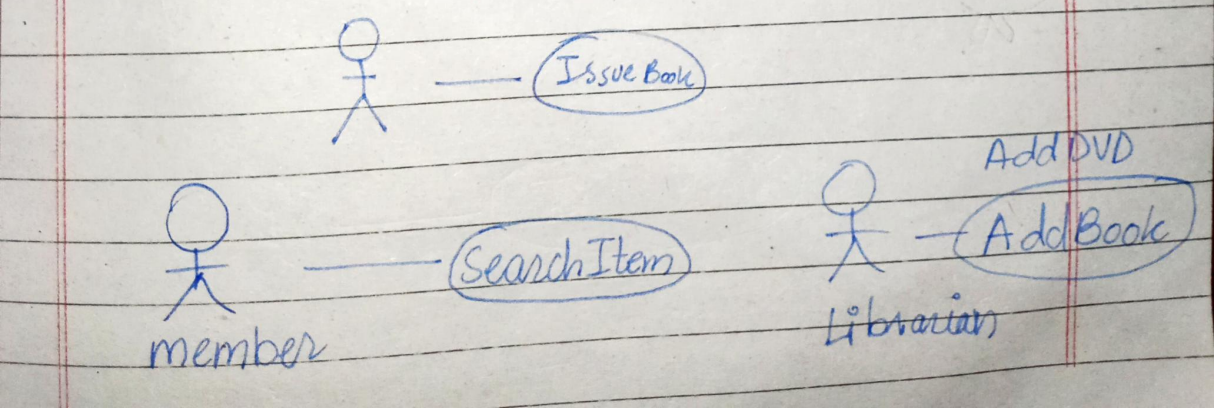
Functions in class diagram

→ Every use case requires one or more functions.

→ The functions should be encapsulated with relevent data.

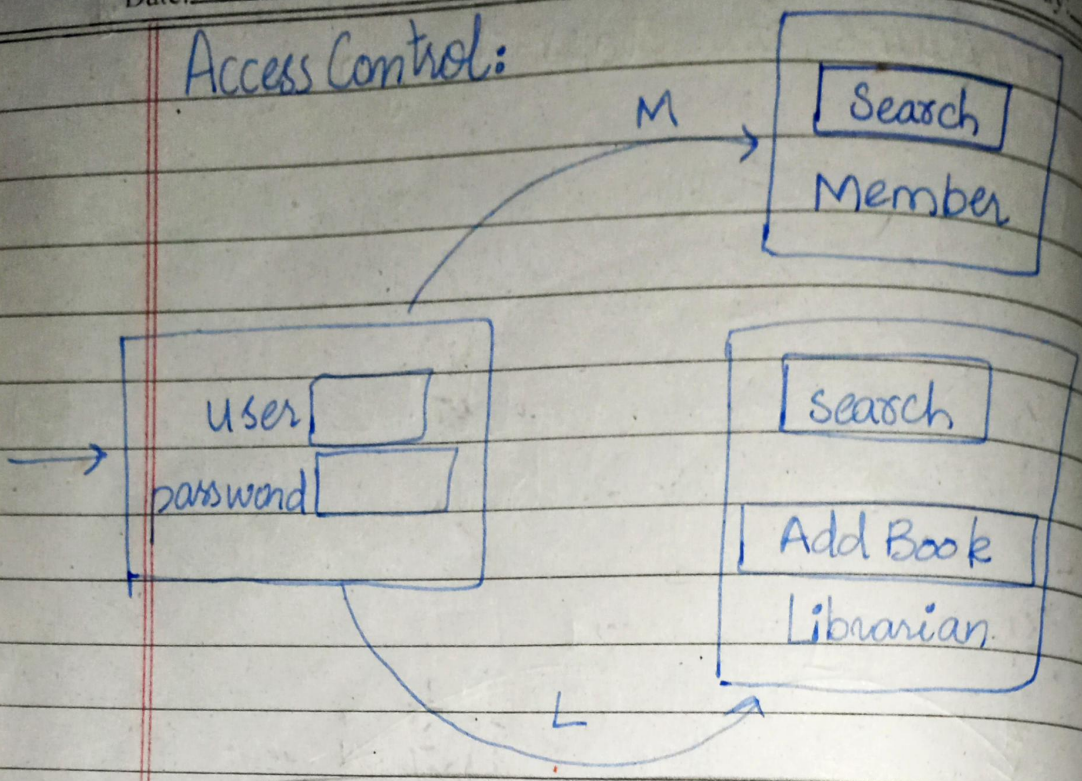


will make 2 different screens.



Date: Day:

Access Control:




```
this.m = m;
this.i = i;   this.date = date
Loan(Member* m, Item* i){
        Data date = currentdate();
    Loan l1 = new loan(m, i);
member* m1 = search(m);
        m1.add(l1);
item* i1 = search(i);
        i1.add(l1);
}
```

issue Book

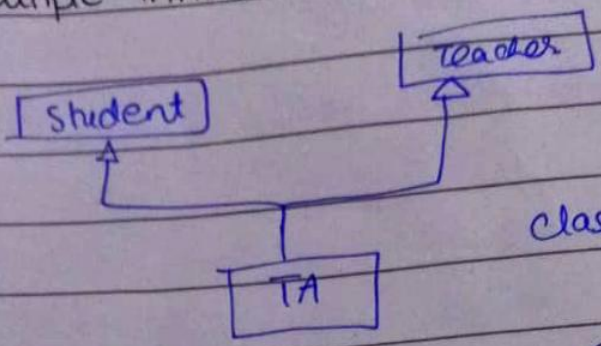
Member ID:

Book ID:

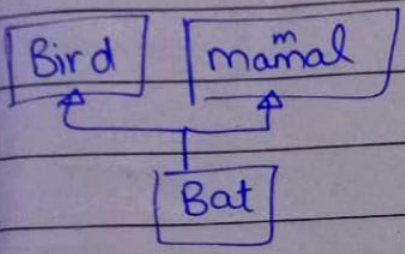
Save

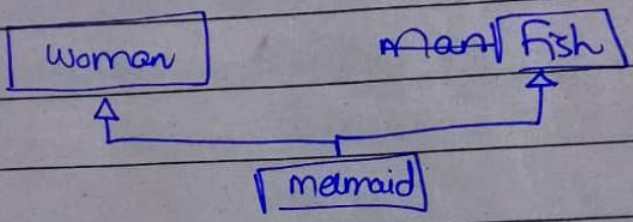
17-09-25 Lecture # 10 Wednesday

Multiple inheritance




```
class TA : public Student,
           public Teacher
{
  ... -
  - - - -
}
```





Problems?

① Student and teacher both has getName() function and when TA calls it compile will give error, not knowing which one to call

```
int main ( )
{ TA. ta1(....);
    ta1.getName( );      // will give error
```

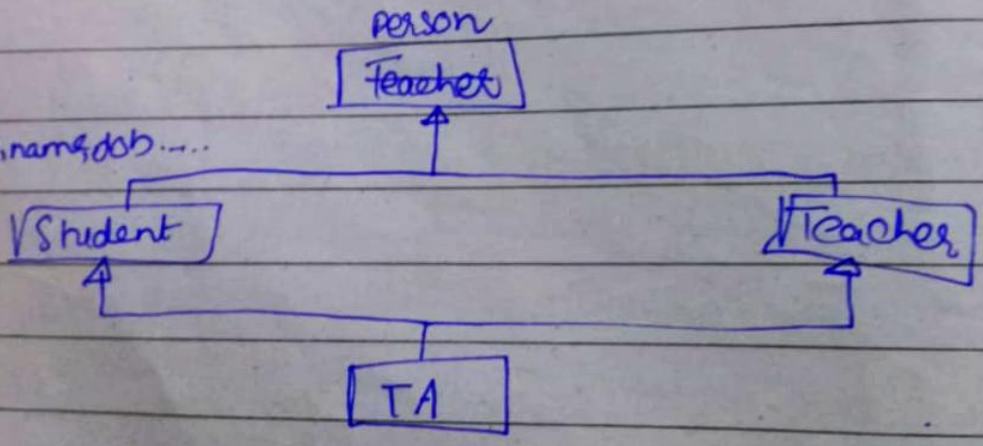
Solution:

Overriding in child class.

```
Class TA: public Student, public Teacher
{ public:
    char* getName()
    { return Student::getName(); }
};
```

// we decide if we want to call teacher's or student's or both and do some concatenation whatever.

Now lets make full diamond (we had half)



Problem: All these variables will be copied twice in TA class (Diamond problem)

Student Teacher

In cout << name will give error.

TA class'

```
class TA : ... .
{ public :
  void foo ( )
  {
    student :: id = 1 ;
    cout << Teacher :: id ;
                          //junk value;
  }
}
```

Problem : Diamond problem

Solution is virtual inheritance. (not polymorphism)

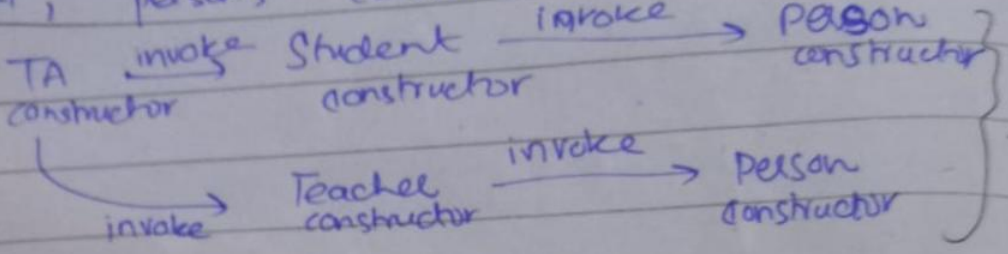
```
class student : public virtual Person
{
    ... ---- -
}
class teacher : public virtual Person
{


}
```

// basically all middle classes k ~~with~~ sath virtual add krn

After adding virtual, we will get single copy in TA class.

Without virtual, we make object of TA, person's constructor is called 2 times.



With virtual, its only called one time.

Java Interfaces

→ we cant do multiple inheritance in Java
A class can only have one parent.
But multiple interfaces are allowed

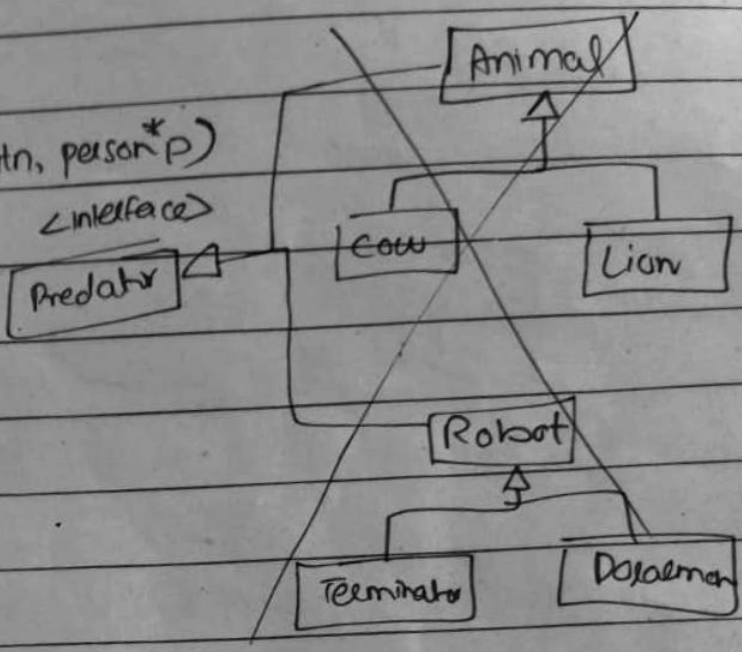
→ interface is A class without data members.
interface can only have functions not variables

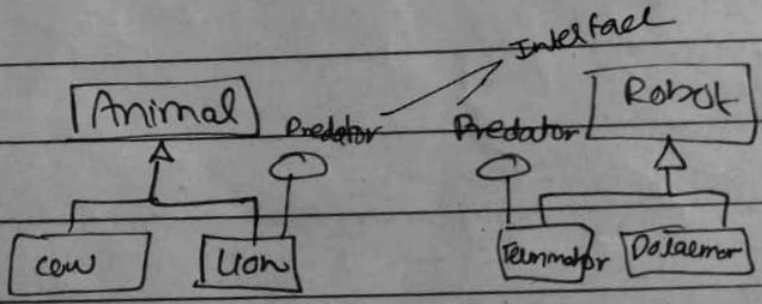
→ Java says a class can only have one parent and multiple interfaces.

→ To implement it in CPP we can make classes w/o variables.

```
interface Predator
{
    void chase (person * p) { ..... }
    void kill (person * p)  { ..... }
}

void foo (Predator *a[ ], int n, person *p)
{
    for(i=0; i<n; i++)
        a[i] ->chase(p);
}
```





Another ways to represent